

TP : création de beans session sans et avec état

1 - Bean Stateless

Ecrire une interface qui propose une méthode salut, prenant une chaîne de caractères en paramètre, et retournant une chaîne de caractères.

Implémenter cette interface dans un bean stateless : la méthode retourne un message de salutation pour la chaîne passée en paramètre (le nom de la personne à saluer).

2 - Bean Stateful

Ecrire une interface qui propose la précédente méthode salut, ainsi qu'une autre méthode re_salut sans paramètre.

Implémenter cette interface dans un bean stateful : la méthode re_salut retourne un message de salutation pour le nom transmis précédemment.

TP : création d'un bean entité

La base de données nommée coursSN contient la table et les données suivantes :

```
create table livre ( isbn char(4) not null primary key,  
                    titre char(20),  
                    dispo smallint default 1 not null, check (dispo in (0,1)));
```

```
insert into livre values('111', 'Le petit prince', 1 );  
insert into livre values('222', 'Barjavel',1);  
insert into livre values('333', 'Le rouge et le noir',1);  
insert into livre values('444', 'Le pere Goriot',1);  
insert into livre values('555', 'Le tour du monde',1);  
insert into livre values('666', 'Vendredi',1);
```

1 - Ecrire un Entity Bean Livre qui gère un livre. Définir les variables d'instance, et un constructeur à 2 arguments : l'isbn et le titre. A la création, le livre est disponible.

2 - Ecrivez une interface GestionLivre qui propose une méthode nouveauLivre à 2 paramètres. Ecrire son implémentation en un bean Stateless.

3 - Ecrivez le fichier de configuration de la persistance, créer l'archive puis déployez.

4 - Ecrivez une application cliente qui crée et supprime des beans.

5 - Transformer le bean de façon à ce qu'on puisse emprunter ou rendre des livres.

Consulter la base de données sur pedago02a :

Pour exécuter des requêtes SQL (et donc consulter le contenu des tables)

```
psql coursSN -U etdsn -h localhost
```

Le client

Il faut copier l'archive jar dans le répertoire du client.

Si le nom de la classe client est Client, et le nom de l'archive serveur TP.jar :

Compilation du client : `javac -classpath TP.jar Client.java`

Exécution du client : `java -classpath $CLASSPATH:TP.jar Client`

Configuration de Glassfish pour accéder à une base de données

- Téléchargez le driver jdbc suivant avant de démarrer le serveur (ou redémarrez le après téléchargement) et le placer dans le répertoire lib de votre domaine (dans domains) :

<http://jdbc.postgresql.org/download/postgresql-9.1-902.jdbc4.jar>

- Pour la configuration voir :

<http://www.hildeberto.com/2010/02/creating-a-connection-pool-to-postgresql-on-glassfish-v3.html>

Le serveur est pedago02a, le port 5432, la base coursSN, l'utilisateur etdsn et son mdp 3tud14nt.

Si vous rencontrez des problèmes de configuration des ressources JDBC, exécutez en ligne de commande : (attention la première commande est sur une seule ligne !)

```
asadmin create-jdbc-connection-pool --datasourceclassname \
org.postgresql.ds.PGSimpleDataSource --restype javax.sql.ConnectionPoolDataSource -- \
property portNumber=5432:password=3tud14nt:user=etdsn:serverName=pedago02a.univ- \
avignon.fr:databaseName=coursSN CoursSNPool \
asadmin create-jdbc-resource --connectionpoolid CoursSNPool jdbc/coursSN
```

Configuration des propriétés de l'unité de persistance

1 - Le fichier persistence.xml a pour en-tête :

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  version="1.0" xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
  http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">
```

2 - Il faut définir les propriétés suivantes :

"javax.persistence.jdbc.driver"	"org.postgresql.Driver"
"javax.persistence.jdbc.url"	"jdbc:postgresql://192.168.2.130:5432/coursSN"
"javax.persistence.jdbc.user"	"etdsn"
"javax.persistence.jdbc.password"	"3tud14nt"